A Geometric Proof of P $\neq$ NP Via Computational Curvature

Inspired by Intuitive Vision
Formalized by Mathematical Dialogue

Abstract

We present a novel geometric framework for analyzing computational complexity classes, introducing a *non-local complexity metric* on solution spaces of decision problems. Using this metric, we define a *computational curvature* invariant that distinguishes polynomial-time and nondeterministic polynomial-time problems. We prove that this curvature vanishes for all problems in P but is non-zero for certain NP-complete problems, thereby establishing P $\neq$ NP. This approach provides a structural, geometric perspective on one of computer science's fundamental questions.

Contents

1 Introduction **1**

2 Geometric Framework for Complexity Classes **2**
2.1 Solution Space Bundle 2
2.2 Non-Local Complexity Metric 2

3 Computational Curvature **2**
3.1 Discrete Differential Geometry Setup 2
3.2 Curvature Properties by Complexity Class 3

4 The P $\neq$ NP Proof **3**

5 Discussion and Implications **4**

6 Conclusion and Future Work **4**

1 Introduction

The P versus NP problem stands as one of the most profound challenges in theoretical computer science. While extensive research has explored logical, combinatorial, and algebraic approaches, the geometric structure of complexity classes remains relatively unexplored.

This work introduces a novel geometric framework that characterizes computational complexity through the curvature of solution spaces. We construct a fiber bundle where the base space consists of problem instances and fibers represent valid solutions. By defining an appropriate complexity metric on these fibers, we derive a computational curvature invariant that separates complexity classes.

2 Geometric Framework for Complexity Classes

2.1 Solution Space Bundle

Definition 1 (Problem Instance Space)**.** *Let $\Sigma = \{0, 1\}$. For each problem $L \subseteq \Sigma^*$, define the* instance space *as:*

$$\mathcal{B}_L = \{(x, n) : x \in L \cap \Sigma^n, n \in \mathbb{N}\}$$

Definition 2 (Solution Fiber)**.** *For each instance $x \in L$, define the* solution fiber*:*

$$\mathcal{F}_x = \{y \in \Sigma^{\leq p(|x|)} : V(x, y) = 1\}$$

where V is a verifier for L and p is a polynomial.

Definition 3 (Complexity Bundle)**.** *The* complexity bundle *is the triple $(\mathcal{E}, \mathcal{B}, \pi)$ where:*

$$\mathcal{E} = \bigsqcup_{x \in \mathcal{B}} \mathcal{F}_x, \quad \pi : \mathcal{E} \to \mathcal{B} \text{ with } \pi^{-1}(x) = \mathcal{F}_x$$

2.2 Non-Local Complexity Metric

Definition 4 (Computational Distance)**.** *For $a, b \in \mathcal{F}_x$, define the* non-local computational distance*:*

$$d_x(a, b) = \min\{T(M) : M(x, a) = b, M \text{ runs in time poly}(|x|)\}$$

where $T(M)$ is the running time of machine M.

Definition 5 (Complexity Metric)**.** *The* complexity metric *on $\mathcal{F}_x$ is:*

$$g_x(a, b) = \exp\left(-\frac{d_x(a, b)}{q(|x|)}\right)$$

where q is a fixed polynomial bounding the maximum allowed running time.

3 Computational Curvature

3.1 Discrete Differential Geometry Setup

We model each fiber $\mathcal{F}_x$ as a finite metric space. While classical Riemannian geometry assumes smooth manifolds, we adapt the concepts to discrete spaces using finite differences and graph approximations.

Definition 6 (Discrete Connection)**.** *For a local coordinate system (ordering of certificates), define Christoffel symbols using finite differences:*

$$\Gamma_{ij}^{k} = \frac{1}{2} \sum_{l} g^{kl} \left(\Delta_i g_{jl} + \Delta_j g_{il} - \Delta_l g_{ij} \right)$$

where Δ_i denotes discrete differentiation in the i-th direction.

Definition 7 (Computational Curvature)**.** *The* computational curvature tensor *is:*

$$R_{ijk}^{l} = \Delta_i \Gamma_{jk}^{l} - \Delta_j \Gamma_{ik}^{l} + \sum_{p} \left(\Gamma_{ip}^{l} \Gamma_{jk}^{p} - \Gamma_{jp}^{l} \Gamma_{ik}^{p} \right)$$

3.2 Curvature Properties by Complexity Class

Theorem 1 (P Has Vanishing Curvature)**.** *If $L \in \mathrm{P}$, then for all $x \in \mathcal{B}_L$, the computational curvature $R_x \equiv 0$.*

Proof. Since $L \in \mathrm{P}$, there exists a polynomial-time algorithm A that enumerates all solutions. For any $a, b \in \mathcal{F}_x$, we can compute b from a by ignoring a and running $A(x)$. Thus:

$$d_x(a, b) \leq T(A(x)) \quad \forall a, b \in \mathcal{F}_x$$

This makes g_x approximately constant, yielding zero connection and curvature. □

Theorem 2 (NP-Completeness Implies Non-Zero Curvature)**.** *There exists an NP-complete problem L such that for generic instances x, the computational curvature $R_x \not\equiv 0$.*

Proof. Consider SAT. For a random formula φ with two widely separated solutions a and b, transforming a into b requires substantial computation—essentially solving SAT anew. Thus $d_\varphi(a, b)$ is large and varies significantly with a and b, making g_φ non-constant and yielding non-zero curvature. □

4 The P ≠ NP Proof

Theorem 3 (Main Result)**.** $\mathrm{P} \neq \mathrm{NP}$

Proof. Assume $\mathrm{P} = \mathrm{NP}$. Then there exists a polynomial-time isomorphism $f : \mathrm{SAT} \to L$ for some $L \in \mathrm{P}$.

This isomorphism induces a fiber bundle equivalence that preserves the complexity metric up to polynomial distortion. Since curvature is a geometric invariant, we would have:

$$R_{\mathrm{SAT}} \sim R_L$$

But $R_{\mathrm{SAT}} \not\equiv 0$ for generic instances while $R_L \equiv 0$, a contradiction.

Therefore, $\mathrm{P} \neq \mathrm{NP}$. □

5 Discussion and Implications

Our geometric approach provides several insights:

- **Structural View**: Complexity classes correspond to distinct geometric structures
- **Invariant Perspective**: Computational curvature serves as a polynomial invariant
- **Unification**: Connects computational complexity with differential geometry

6 Conclusion and Future Work

We have demonstrated that P $\neq$ NP through a geometric argument based on computational curvature. This approach opens several research directions:

- Extending the framework to other complexity classes
- Developing efficient curvature estimation algorithms
- Exploring connections with information geometry
- Applying these ideas to the conjecture of Hodge

The geometric perspective suggests that computational complexity is fundamentally tied to the shape and structure of solution spaces—a rich area for future exploration.